Amendment to the specification

Please make the following changes to the specification. The page/line references are to the original specification. All amendments made in prior responses are incorporated into the text of the following further-amended paragraphs as if they were in the original text. Only the new changes are indicated below, wherein bracketed text is removed and underlined text is newly added. None of the amendments add new matter, but rather clarify certain descriptions already included in the specification.

Please replace the paragraph beginning at page 7, line 3 with the following:

FIG. 2 schematically illustrates the contents of the mapping table 200. As described above, the table 200 contains entries 210 (rows) that [include] indicate a mapping between one or more virtual drive segments [220] 222 of a virtual disk 220 and storage locations 230 on the storage devices. The storage locations 230 identify the particular storage device and part of the storage device, which correspond to the virtual disk 150 index. The form for the storage locations must be appropriate for the storage network being used. In a SCSI network, each of the storage locations 230 includes a LUN identifier 233 and a block identifier 235, also called an offset. All of the other fields in a mapping table entry 210 are simple integers or binary state values.

Please replace the paragraph beginning at page 7 line 14 with the following:

This disclosure describes the mapping table 200 as having one entry 210 per each "disk block" of virtual disk 220. While possible to build, this would result in huge mapping tables and highly fragmented mapping, both of which introduce undesirable performance degradations. In another implementation, each mapping table entry 210 represents a variable sized group of contiguous virtual disk blocks that map to contiguous blocks on one of the physical storage devices. This configuration of the mapping table 200 offers mapping flexibility and dense mapping structures, but introduces greater algorithmic complexity in managing the variable sized blocks and greater map entry lookup costs. Therefore, the table 200 may use mapping table entries 210, each having a fixed size number of contiguous blocks ("segments") on the virtual disk 150 that map to one storage device. While this configuration for the table 200 is possibly not as dense as variable sized block mapping, the configuration offers the simplest and highest performance map access and space management. In this configuration, each of the entries 210 contains a virtual disk segment [220] 222 instead of a virtual disk block. Regardless of the specifics of the table 200, the table 200 must map a virtual drive segment [220] 222 to each physical storage block involved in I/O operations.

Please replace the paragraph beginning at page 8 line 19 with the following:

The disclosure first describes the states prior to explaining some of the functions for the states. The table 200 generally includes at least two states: (1) an invalid state 240 indicating whether any I/O operations may occur on the virtual disk segment [220] 222 and the corresponding physical location 230; and (2) a no-write (Nw) state 250 indicating whether the data contained at the corresponding physical location 230 may be changed. The invalid state 240

and the Nw state 250 are particularly important in allowing dynamic loading of mapping table entries, dynamic mapping changes, volatility of mapping table entries, and data sharing among similar virtual disks 150.

Please replace the paragraph beginning at page 9 line 26 with the following:

As presented above, the Nw state 250, when activated, indicates that any write operations to the virtual disk segment(s) [220] 222 represented by the entry 210 cause the agent 110 to send a fault message the controller 120. The agent 110 does not allow the host 140 to write to the storage locations 230 until the controller 120 returns a fault response to deactivate the Nw state 250. Unlike the invalid state 240, the activated Nw state 250 does not prevent read operations from generating faults. Instead, the agent 110 generally allows the host 140 to proceed to access data at the storage location 230. Accordingly, if only the Nw state is activated, the mapping table entry 210 must contain a useable storage location 230.

Please replace the paragraph beginning at page 10 line 9 with the following:

In another configuration, the mapping table 200 further includes a zero (Z) state 260. When active, the Z state 260 indicates that the virtual disk segment [220] 222 represented by the entry 210 contains all zero bytes. This feature allows a virtual disk 150 to be created and gives the virtual disk 150 the appearance of being initialized without the need to allocate or adjust any underlying non-virtual storage. If an entry 210 contains an active Z state 260, the agent 110 ignores the storage address 230. If the host 140 attempts to read information stored at storage address 230, the agent 110 returns only zero-filled blocks regardless of the actual contents of the

storage address 230. On the other hand, any attempts to write data at the storage address 230 when Z state 260 is activated cause the agent 110 to send a fault message to the controller 120. The agent 110 does not allow the host 140 to write to the storage locations 230 until the controller 120 returns a fault response that deactivates the Z state 260.

Please replace the paragraph beginning at page 11 line 22 with the following:

The map fault message from the agent 110 generally identifies the requested I/O operation, the virtual disk segment [220] 222 involved, and the table state preventing the I/O operation. After a fault occurs, the agent does not attempt to carry out the I/O operation. Instead, the controller 120 uses the fault message to select the proper response to the faulted I/O operation, (e.g. load map entry, change map entry, delay until some other operation has completed). The controller 120 response informs the mapping agent 110 how to proceed to overcome the cause for the fault.

Please replace the paragraph beginning at page 14 line 9 with the following:

As illustrated in FIG. 3, in the context of a distributed table-driven virtual storage network, such as the above-described virtual storage system 100, a stored record of the contents of the virtual drive can be preserved by modifying the mapping table 200 to prevent any changes to the table entries 210 or to the data stored in the corresponding storage locations 230. This may be accomplished in table 200 by activating the Nw state 250 for any and all of the table entries 210 that map virtual disk blocks or segments [220] 222 storage locations 230.

Please replace the paragraph beginning at page 14 line 17 with the following:

The activation of the Nw state 250 for any and all of the table entries 210 is generally accomplished in the system 100 according to the following description of a virtual disk or a snapshot disk creation 300 operation, e.g., creation of a new snapshot disk that generally involves copying the contents of a previously created mapping table but not moving data to allow the new snapshot disk to share the same storage as the original disk, which is space efficient and fast. The disk copy 300 operation begins at step 305. In step 310, the controller 120 activates the Nw state 250 for all mapping table entries 210 in the persistent copy of the mapping table 200 for the original disk. The controller uses a set-entry-state command to communicate this change to all of the mapping agents 110 that map to this virtual disk 150 by setting the Nw state 250 for all mapping table entries 210 in these mapping agents 110, step 320. After this point, all attempts to write to the virtual disk 150 in the table 200 generate mapping faults to the controller 120. Alternatively, if the Nw state is not set, step 315, the controller 120 may activate the invalid flag 240 for all the mapping agent 110 map entries, step 325. The use of invalid flag 240 instead of the Nw flag 250 generates mapping faults for read operation that are otherwise allowed during the period when the Nw state 250 is activated. The key concept is that, at a minimum, all write attempts through the table 200 generate faults.

Please replace the paragraph beginning at page 15, line 5 with the following:

As described above, the controller 120 set_entry_state signals to the mapping agents 110 to activate the blocking flag that blocks the controller from initiating the table copy until prior I/O operations that were in progress against virtual disk segment 222 at the time of the

set_entry_operation have completed. As a result, the mapping agent 110 allows all prior I/O operations to complete prior to responding to the controller 120 and implementing the changes to the Nw state 250. In this way, the controller 120 can know when all outstanding writes to the original disk are completed. The controller 120 then copies the entire contents of the mapping table 200 for the original disk to a new mapping table 200 for the snapshot disk, step 330. This step 330 includes copying the active Nw state 250 for the table entries 210, so that later attempts to write to the snapshot disk containing the copy also generate mapping faults to the controller 120.

Please replace the paragraph beginning at page 17 line 21 with the following:

The controller 120 then updates its persistent copy of the mapping table 200 for all the virtual disks 150 that share the faulting segment, except for the mapping table that maps the particular virtual disk 150 associated with the I/O fault, step 550. In particular the controller 120 remaps the virtual disk segments [220] 222 to the newly allocated storage location 230. To update the mapping tables, the controller 120 deactivates the Nw state 250 in the persistently stored table. As part of the step 550, the controller 120 changes the storage location 230 to refer to the newly allocated segment.